# BEST AVAILABLE IMAGES

Defective images within this document are accurate representations of the original documents submitted by the applicant.

Defects in the images may include (but are not limited to):

- BLACK BORDERS
- TEXT CUT OFF AT TOP, BOTTOM OR SIDES
- FADED TEXT
- ILLEGIBLE TEXT
- SKEWED/SLANTED IMAGES
- COLORED PHOTOS
- BLACK OR VERY BLACK AND WHITE DARK PHOTOS
- GRAY SCALE DOCUMENTS

## IMAGES ARE BEST AVAILABLE COPY.

**As rescanning documents *will not* correct images, please do not report the images to the Image Problem Mailbox.**

(19) JAPANESE PATENT OFFICE

JPA2002-328787

PATENT ABSTRACTS OF JAPAN

(11) Publication number: **2002328787 A**

(43) Date of publication of application: **15.11.02**

(51) Int. Cl **G06F 3/12**
**B41J 29/00**
**B41J 29/42**

(21) Application number: **2001132929**

(22) Date of filing: **27.04.01**

(71) Applicant: **CANON INC**

(72) Inventor: **MINAGAWA TOMONORI**

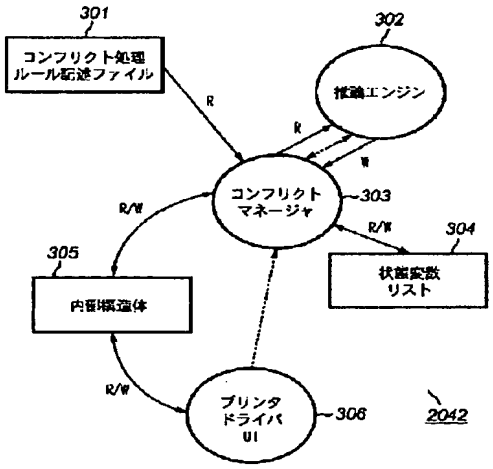
# (54) USER INTERFACE CONTROLLER AND USER INTERFACE CONTROL METHOD



(57) Abstract:

PROBLEM TO BE SOLVED: To provide an inexpensive and high quality user interface controller and its method by sharing a conflict processing module to data inputted through a user interface(UI) by conflict processing to data inputted without using the UI and the matching processing of the whole data.

SOLUTION: During conflict processing, a state variable list (304) having set values or statuses for the respective set items of a user interface(UI) is placed in a work area, and the validity/invalidity of the set items can be easily judged even at UI non-display processing by performing access to the state variable list (304) instead of an internal structural body (305). Also, at realizing the matching of the whole data, a starting point list in which the set items are listed in the priority order is read, and the conflict processings are successively performed so that mismatching between the set items can be prevented, and the matching of the whole data can be realized.



JPA2002-328787

(19)日本国特許庁（ＪＰ）　(12)**公開特許公報（A）**　(11)特許出願公開番号
特開2002－328787
（P2002－328787A）
(43)公開日　平成14年11月15日(2002.11.15)

| (51)Int.Cl.[7] | 識別記号 | ＦＩ | | テーマコード(参考) | |
|---|---|---|---|---|---|
| Ｇ０６Ｆ　3/12 | | Ｇ０６Ｆ | 3/12 | Ｃ | ２Ｃ０６１ |
| Ｂ４１Ｊ　29/00 | | Ｂ４１Ｊ | 29/42 | Ｆ | ５Ｂ０２１ |
| 29/42 | | | 29/00 | Ｔ | |

審査請求　未請求　請求項の数６　ＯＬ　（全 12 頁）

(21)出願番号　特願2001－132929(P2001－132929)

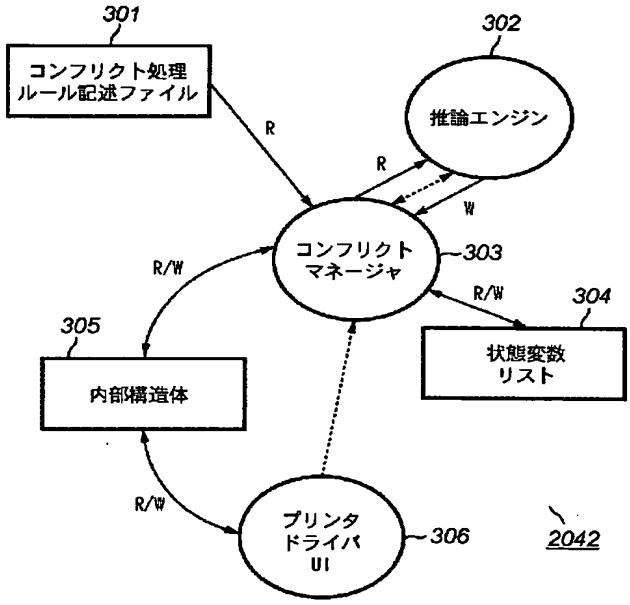
(22)出願日　平成13年４月27日(2001.4.27)

(71)出願人　000001007
キヤノン株式会社
東京都大田区下丸子３丁目30番２号
(72)発明者　皆川　智徳
東京都大田区下丸子３丁目30番２号　キヤノン株式会社内
(74)代理人　100076428
弁理士　大塚　康徳　(外３名)

Ｆターム(参考)　2C061 AP01 CQ05 CQ14 CQ29
5B021 AA01 BB01 CC05

(54)【発明の名称】　ユーザインタフェース制御装置および方法

(57)【要約】
【課題】　ユーザインタフェース（UI）を介して入力されるデータに対するコンフリクト処理モジュールを、UIを介さずに入力されるデータに対するコンフリクト処理とデータ全体の整合処理とで共用可能にし、低コストで高品質なユーザインタフェース制御装置および方法を提供すること。
【解決手段】　コンフリクト処理中は、UIの各設定項目毎に設定値やステータスを持つ状態変数リスト(304)を作業領域に置き、内部構造体(305)のかわりに状態変数リスト(304)にアクセスすることで、UI非表示の処理の際にも設定項目の有効／無効判定を容易に行うことを可能にする。また、データ全体の整合を取る際には、優先順位をつけて設定項目を列挙した起点リストを読み込んで順次コンフリクト処理を実施することで設定項目間の不整合を解消し、データ全体の整合をとる。



【特許請求の範囲】

【請求項１】　所定の被制御装置に対する設定変更要求をユーザインタフェースを介して入力する第１の入力手段と、前記被制御装置に対する設定変更要求を前記ユーザインタフェースを介さずに入力する第２の入力手段とを有し、前記ユーザインタフェースにおける各設定項目の設定状態の間に生じる不整合を回避するためのユーザインタフェース制御装置であって、

前記ユーザインタフェースにおける各設定項目の、前記設定状態を示す設定情報と直接変更の可否を示す情報とを記憶する第１の記憶領域と、

前記ユーザインタフェースの内部構造体として、前記第１の記憶領域における前記各設定項目の前記設定情報を反映して記憶する第２の記憶領域と、

前記第１または第２の入力手段より入力される前記設定変更要求に応じて適用される、所定の不整合回避戦略を示すコンフリクト処理ルールを記憶する第３の記憶領域と、

前記設定変更要求のあった前記設定項目に対応する前記第１の記憶領域における前記直接変更の可否を示す情報が、直接変更可であることを示しているか否かを判断する判断手段と、

前記判断手段により、前記設定変更要求のあった前記設定項目に対応する前記第１の記憶領域における前記直接変更の可否を示す情報が、直接変更可であることを示していると判断された場合において、前記コンフリクト処理ルールに従って、前記第１の記憶領域に記憶されている特定の前記設定項目の、前記設定情報および／または前記直接変更の可否を示す情報を変更する変更手段と、

を有することを特徴とするユーザインタフェース制御装置。

【請求項２】　前記コンフリクト処理ルールに関係する前記各設定項目を記述したリストを記憶する第４の記憶領域と、

前記リスト中の各設定項目を、前記設定変更要求のあった設定項目とみなして、前記判断手段、前記変更手段における各処理を適用することで、前記各設定項目の設定状態の間の整合をとる手段と、

を更に有することを特徴とする請求項１に記載のユーザインタフェース制御装置。

【請求項３】　前記所定の被制御装置は、画像形成装置であることを特徴とする請求項１または２に記載のユーザインタフェース制御装置。

【請求項４】　所定の被制御装置に対する設定のためのユーザインタフェースにおける各設定項目の設定状態の間に生じる不整合を回避するためのユーザインタフェース制御方法であって、

前記被制御装置に対する設定変更要求を前記ユーザインタフェースを介して入力する第１の入力工程と、

前記被制御装置に対する設定変更要求を前記ユーザインタフェースを介さずに入力する第２の入力工程と、

前記ユーザインタフェースにおける各設定項目の、前記設定状態を示す設定情報と直接変更の可否を示す情報とを記憶している状態変数リストにおける、前記設定変更要求のあった前記設定項目に対応する前記直接変更の可否を示す情報が、直接変更可であることを示しているか否かを判断する判断工程と、

前記判断工程により、前記状態変数リストにおける、前記設定変更要求のあった前記設定項目に対応する前記直接変更の可否を示す情報が、直接変更可であることを示していると判断された場合において、前記第１または第２の入力工程より入力される前記設定変更要求に応じて適用される、所定の不整合回避戦略を示すコンフリクト処理ルールに従って、前記状態変数リスト中に記憶されている特定の前記設定項目の、前記設定情報および／または前記直接変更の可否を示す情報を更新する更新工程と、

を有することを特徴とするユーザインタフェース制御方法。

【請求項５】　コンピュータ装置が実行可能なプログラムであって、当該プログラムを実行したコンピュータ装置を、請求項１ないし３のいずれか１項に記載のユーザインタフェース制御装置として機能させることを特徴とするプログラム。

【請求項６】　請求項５に記載のプログラムを格納した記憶媒体。

【発明の詳細な説明】

【０００１】

【発明の属する技術分野】本発明は、ユーザインタフェースを介して入力される所定の制御対象に対する設定データの間に生じる不整合を回避するユーザインタフェース制御装置および方法に関する。

【０００２】

【従来の技術】ユーザインタフェース（以下、「ＵＩ」ともいう。）を介してユーザから複数の設定値の入力を受け付け、それらの設定値に基づき制御される機器は、数多くある。ホストコンピュータに接続され、そのホストコンピュータにおいて設定された情報に基づき画像形成処理を行う画像形成システム（プリンタシステム）はその一例である。プリンタシステムのホストコンピュータは一般に、印刷動作を制御するいわゆるプリンタドライバ、およびユーザからの印刷設定等を受け付けるＵＩを含む印刷処理関連プログラムを有する。

【０００３】印刷処理関連プログラムは、ＵＩを介してユーザから設定値の入力を受け付ける度に、それまでに設定された複数の設定値の中で関連する設定値の値との関係を評価し、設定値間の不整合（コンフリクト）がないかどうかを判別する。コンフリクトの例としては、記録媒体としてセットされたＯＨＰシートに対して両面印刷を行わせるような設定等のユーザにとって不都合と予

想される設定や、プリンタに不可能な動作を行わせるような設定等がある。

【０００４】コンフリクトがあった場合には、それを回避するためのコンフリクト処理を実施する必要がある。

【０００５】ところで、ユーザインタフェースでの設定を保存する内部構造体に対し、他のプログラムからユーザインタフェースを介さずに（すなわち、ユーザインタフェースを表示することなく）、直接設定値の変更を行う場合がある。この場合にも、設定変更に関連する項目との不整合を評価し、ユーザインタフェースを介してユーザが変更した場合と同様のコンフリクト処理を実施する必要がある。

【０００６】また、デバイスの設定変更やアプリケーションとの連携の過程で内部構造体設定の整合性が失われる場合があり、ユーザインタフェースを開く際や印刷実行前等にはデータ全体の整合処理が必要となる。この場合にも、ユーザがユーザインタフェースを開いて操作を行った場合と同一の整合処理結果が得られる必要がある。

【０００７】

【発明が解決しようとする課題】ユーザインタフェースを表示しないときのコンフリクト処理と、データ全体の整合処理とでは、どちらのケースでもコンフリクト処理ルールは同一である。このことから、コンフリクト処理部をデータ処理部とユーザインタフェース処理部とに分けて、ユーザインタフェースを表示しない場合はデータ処理部のみを流用することが考えられる。

【０００８】しかし、ユーザインタフェース非表示の処理に関しては、ユーザインタフェース表示時はグレイアウト（設定項目を薄いグレイで表示すること）されていて直接変更することができない項目についても、変更要求のイベントが発生し予期しない変更処理が実施される可能性があるため、共通の処理モジュールの利用が困難であった。

【０００９】また、データ全体の整合処理に関しては、ユーザインタフェース表示時では、設定変更された項目を起点として関連する項目との整合を評価し、更新が発生する場合はさらに関連する項目へと連鎖するフローとなるのに対して、データ全体の場合は、設定変更による起点が存在せず、全体にわたるチェックが必要になるという点で前提が異なる。このことからも、共通の処理モジュールの利用が困難であった。

【００１０】しかしながら、これらをそれぞれ独立した処理モジュールにすると、ほとんど同一のアルゴリズムであるにも関わらず、別々の処理コードを作成する手間がかかるうえ、不整合検知処理の追加や変更に伴って各処理での漏れや食違いが出やすいという問題があった。

【００１１】本発明はこのような問題点に鑑みてなされたもので、ユーザインタフェースを介して入力されるデータに対するコンフリクト処理を行うためのモジュールを、ユーザインタフェースを介さずに入力されるデータに対するコンフリクト処理とデータ全体の整合処理とで共用可能にして、各処理ごとの専用のコンフリクト処理モジュールを作成する必要をなくし、もって低コストで高品質なユーザインタフェース制御装置および方法を提供することを目的としている。

【００１２】

【課題を解決するための手段】上記目的を達成するため、例えば本発明のユーザインタフェース制御装置は、以下の構成を備える。すなわち、所定の被制御装置に対する設定変更要求をユーザインタフェースを介して入力する第１の入力手段と、前記被制御装置に対する設定変更要求を前記ユーザインタフェースを介さずに入力する第２の入力手段とを有し、前記ユーザインタフェースにおける各設定項目の設定状態の間に生じる不整合を回避するためのユーザインタフェース制御装置であって、前記ユーザインタフェースにおける各設定項目の、前記設定状態を示す設定情報と直接変更の可否を示す情報とを記憶する第１の記憶領域と、前記ユーザインタフェースの内部構造体として、前記第１の記憶領域における前記各設定項目の前記設定情報を反映して記憶する第２の記憶領域と、前記第１または第２の入力手段より入力される前記設定変更要求に応じて適用される、所定の不整合回避戦略を示すコンフリクト処理ルールを記憶する第３の記憶領域と、前記設定変更要求のあった前記設定項目に対応する前記第１の記憶領域における前記直接変更の可否を示す情報が、直接変更可であることを示しているか否かを判断する判断手段と、前記判断手段により、前記設定変更要求のあった前記設定項目に対応する前記第１の記憶領域における前記直接変更の可否を示す情報が、直接変更可であることを示していると判断された場合において、前記コンフリクト処理ルールに従って、前記第１の記憶領域に記憶されている特定の前記設定項目の、前記設定情報および／または前記直接変更の可否を示す情報を変更する変更手段と、を有することを特徴とする。

【００１３】

【発明の実施の形態】（ハードウェア構成）図１は本発明の一実施形態を示す印刷処理システムのブロック構成図である。印刷処理システムは、ホストコンピュータ３０００と、プリンタ１５００より構成される。

【００１４】ホストコンピュータ３０００において、１はシステムバス４に接続される各デバイスを総括的に制御するＣＰＵ、２はＣＰＵ１の主メモリ、ワークエリア等として機能するＲＡＭである。３は各種プログラム、データを格納するＲＯＭであって、各種フォントを記憶するフォントＲＯＭ３ａ、ブートプログラムやＢＩＯＳ等を記憶するプログラムＲＯＭ３ｂ、および各種データを記憶するデータＲＯＭ３ｃに区分けして構成されている。

【００１５】５はキーボードコントローラ（ＫＢＣ）で、キーボード９や不図示のポインティングデバイス（マウス）からのキー入力を制御する。６はＣＲＴコントローラ（ＣＲＴＣ）であり、ＣＲＴディスプレイ（ＣＲＴ）１０の表示を制御する。

【００１６】外部メモリ１１（ディスクコントローラ（ＤＫＣ）７によりアクセス制御される）は、ハードディスク（ＨＤ）やフロッピー（登録商標）ディスク（ＦＤ）等であり、図示の如く、オペレーティングシステムプログラム（以下ＯＳ）２０５をはじめ各種アプリケーション（例えば、図形、イメージ、文字、表等が混在した文書処理を行う文書処理アプリケーションプログラム）２０１、印刷処理関連プログラム２０４を記憶する他、ユーザファイル、編集ファイル等も記憶する。印刷処理関連プログラム２０４は、ページ記述言語を用いて記述される印刷データを生成するプログラムであって、同系列の複数のプリンタに対して共通に利用されうる。また、プリンタ制御コマンド生成モジュール（以下「プリンタドライバ」という）２０４１、プリンタドライバＵＩ制御モジュール２０４２を含む。

【００１７】８はプリンタコントローラ（ＰＲＴＣ）であり、双方向性インタフェース２１を介してプリンタ１５００に接続され、プリンタ１５００との通信制御処理を行う。

【００１８】外部メモリ１１に記憶されたアプリケーションは、ＲＡＭ２にロードされてＣＰＵ１により実行されることになる。また、ＣＰＵ１は、例えばＲＡＭ２へのアウトラインフォントの展開（ラスタライズ）処理を実行し、ＣＲＴ１０上でのWYSIWYG（What you see is What you get）を可能としている。さらに、ＣＰＵ１は、ＣＲＴ１０上の不図示のマウスカーソル等で指示されたコマンドに基づいて登録された種々のウインドウを開き、種々のデータ処理を実行する。ユーザは印刷を実行する際、印刷設定設定画面（プリンタドライバＵＩ制御モジュール２０４２により制御される）を開き、プリンタの設定や、印刷モードの選択を含むプリンタドライバ２０４１に対する印刷処理の設定を行うことができる。

【００１９】プリンタ１５００において、１２はプリンタ１５００の全体を制御するＣＰＵである。１９はＣＰＵ１２の主メモリ、ワークエリア等として機能するとともに、出力情報展開領域、環境データ格納領域、ＮＶＲＡＭ等に用いられるＲＡＭであり、図示しない増設ポートに接続されるオプションＲＡＭによりメモリ容量を拡張することができるように構成されている。１３はＲＯＭであり、各種フォントを記憶するフォントＲＯＭ１３ａ、制御プログラム等を記憶するプログラムＲＯＭ１３ｂ、および各種データを記憶するデータＲＯＭ１３ｃより構成される。

【００２０】外部メモリ１４（メモリコントローラ（ＭＣ）２０によりアクセスを制御される）は、オプションとして接続されるハードディスク（ＨＤ）、フロッピーディスク（ＦＤ）、ＩＣカード等であり、フォントデータ、エミュレーションプログラム、フォームデータ等を記憶する。ハードディスク等の外部メモリ１４が接続されていない場合には、ＲＯＭ１３のデータＲＯＭ１３ｃに、ホストコンピュータ３０００で利用される情報等を記憶することになる。なお、外部メモリ１４は１個に限らず、複数備えるものであってもよく、例えば、内蔵フォントに加えてオプションフォントカード、言語系の異なるプリンタ制御言語を解釈するプログラム等を格納した外部メモリを複数接続できるように構成されていてもよい。

【００２１】操作部１５０１にはユーザからの操作を受け付ける操作パネルが設けられ、その操作パネルには操作のためのスイッチおよびＬＥＤ表示器等が配されている（図示は省略）。また、図示しないＮＶＲＡＭを有し、操作パネル１５０１からのプリンタモード設定情報を記憶するようにしてもよい。

【００２２】プリンタＣＰＵ１２は、ＲＯＭ１３のプログラムＲＯＭ１３ｂに記憶された制御プログラム等に基づき、印刷部インタフェース１６を介してシステムバス１５に接続される印刷部（プリンタエンジン）１７に出力情報としての画像信号を出力する。また、ＣＰＵ１２は入力部１８を介してホストコンピュータ３０００との通信処理が可能となっており、プリンタ１５００内の情報等をホストコンピュータ３０００に通知可能に構成されている。

【００２３】（ソフトウェア構成）図２は、所定のアプリケーションおよびプリンタ１５００を制御対象とする印刷処理関連プログラムを起動して、ホストコンピュータ３０００上のＲＡＭ２にロードされた状態のＲＡＭ２のメモリマップを示している。ＲＡＭ２には、図示の如く、ＢＩＯＳ２０６、ＯＳ２０５をはじめ、アプリケーション２０１、印刷処理関連プログラム２０４、および関連データ２０３がロードされており、さらに空きメモリ領域２０２も確保されている。これにより、アプリケーション２０１および印刷処理関連プログラム２０４は実行可能状態にある。

【００２４】印刷処理関連プログラム２０４におけるプリンタドライバＵＩ制御モジュール２０４２は、ユーザによる印刷設定指令に応じてＣＲＴ１０に印刷設定画面を表示しユーザからの設定を可能にする。

【００２５】図８に、ＵＩとしての印刷設定画面の表示例を示す。同図において、[Print Style]欄８０は、印刷レイアウトを指定する欄であり、ユーザは例えば、片面印刷（1-Sided Printing）８０１、両面印刷（2-Sided Printing）８０２、および製本印刷（Booklet Printing）８０３のコントロールうちの、いずれかに対応するラジオボタンをマウスでクリックすることで指定するこ

とができる。

【００２６】[Finishing]欄８１は、印刷済み記録媒体の出力順序および仕上げについて指定する欄であり、ユーザは例えば、以下のいずれかから指定可能となっている。

【００２７】[Collate] ８１１

部単位印刷。Ｎページの文書をＭ部印刷する場合に、第１ページ、第２ページ、…、第Ｎページの順に１枚ずつ出力し、これをＭ回繰り返す。

【００２８】[Group] ８１２

ページ単位印刷。Ｎページの文書をＭ部印刷する場合に、第１ページをＭ枚、第２ページをＭ枚、…、第Ｎページをｍ枚、の順に出力する。

【００２９】[Staple] ８１３

ステープル仕上げ。上記[Collate] ８１１と同様に部単位で出力し、仕上げとして各部ごとにステープラで止める。

【００３０】なお、本明細書では、上記したようなユーザ設定可能項目を「プリンタ機能」または単に「機能」ともよぶ。この他にも多くのプリンタ機能を有するが、説明を簡単にするため省略する。

【００３１】ここで、ユーザにとって不都合と思われる設定の組み合わせ、意味のない設定の組み合わせ、すなわち設定値間の不整合（コンフリクト）は、プリンタドライバＵＩ制御モジュール２０４２により、以下詳細に説明するコンフリクト処理として回避されるように設計される。

【００３２】図３は、実施形態における印刷処理関連プログラム２０４のプリンタドライバＵＩ制御モジュール２０４２の概略構成を示している。３０３は、各モジュール間のデータの受け渡しやデータの更新等を管理してコンフリクト処理を統括するコンフリクトマネージャである。３０６が、上記した印刷設定画面表示制御を行い、本プリンタドライバＵＩ制御モジュール２０４２におけるメインプログラムとしてのプリンタドライバＵＩである。３０１は、後述する記述形式で記述される不整合回避戦略を示すコンフリクト処理ルールを列記したコンフリクト処理ルール記述ファイルである。３０２は、コンフリクト処理ルール記述ファイル３０１をロードして、入力された設定値に対してコンフリクト処理ルールを適用し、各機能の状態を推論する推論エンジン、３０４は、各プリンタ機能の状態をリスト形式で表示する状態変数リストであり、ユーザからの入力およびコンフリクト処理ルール記述ファイル３０１の内容に基づき更新されうる。３０５は、プリンタドライバＵＩ３０６が提供する画面表示の基になる帳票としての内部構造体であり、状態変数リスト３０４と連動して各プリンタ機能の状態を所定の形式で表示する。

【００３３】その後に、プリンタドライバＵＩ３０６を介してユーザからの設定情報を受け取ったコンフリクトマネージャ３０３は、コンフリクト処理ルール記述ファイル３０１を参照する。このことは、図示の如くコンフリクト処理ルール記述ファイル３０１からコンフリクトマネージャ３０３に向かう矢印で、「R(Read)」として表示されている。参照の結果、設定情報がコンフリクト処理ルールに適合する場合、コンフリクト処理が適用される。そうして、コンフリクトマネージャ３０３は、状態変数リスト３０４および内部構造体３０５を更新してプリンタドライバＵＩ３０６に反映させる。この更新作業は、図示の如くコンフリクトマネージャ３０３は、状態変数リスト３０４および内部構造体３０５とは各々双方向矢印で結ばれ、「R/W (Read/Write)」として表示されている。

【００３４】図４は、図３に示した各モジュールで扱われるデータの関連を説明する模式図である。図４において、コンフリクト処理ルール記述ファイル３０１は、推論エンジン３０２にインクルード（ロード）されたかたちで参照される。このコンフリクト処理ルール記述ファイル３０１はコンフリクトマネージャ３０３にも参照され、状態変数リスト３０４はコンフリクト処理ルールに従った内容を有することになる。また、内部構造体３０５と状態変数リスト３０４とは先に述べたとおり連動表示されるものであるから、互いにマッピングされる関係にある。そしてこの状態がプリンタドライバＵＩ３０６の制御によってユーザに見えるかたちで表現される。

【００３５】なお、図４における状態変数リスト３０４は、各プリンタ機能（同図中「キー」の欄A, B, C,...）毎に状態値（ON/OFF）だけが簡略的に記述されているが、実際にはこの他に例えば、対応するコントロールのステータス（有効／無効）、フラグ、ＵＩの表示更新メソッド、内部構造体３０５からのデータ取得メソッド、内部構造体３０５へのデータ反映メソッド等が記述されることになる。なお、「メソッド」とは、目的の機能を果たすためのデータ処理を記述したプログラムのことをいう。

【００３６】内部構造体３０５は、各プリンタ機能の状態を例えば、

データ型　機能名メンバ　状態メンバ

の構文で表し、プリンタ機能名A, B, C に対応する機能名メンバをそれぞれ cA,cB, cC のように表現する。そして、各機能に対する状態メンバは、状態変数リスト３０４における状態値がマッピングされている。

【００３７】（コンフリクト処理ルールの記述形式）次に、コンフリクト処理ルール記述ファイル３０１について説明する。

【００３８】各ルールは、論理（ロジック）を用いて数学的に形式化される。述語は、

プリンタ機能名（引数）

の形で記述される。引数としては、ON/OFFの他、数値が入る場合もある（例えば、印刷部数等）。左辺には「プ

リンタ機能名（引数）」を、右辺には左辺が成り立つための論理条件を記述し、記号「←」で関係づける。例えば、

A(ON) ← B(ON).

は、「プリンタ機能Bの状態がONのときは、プリンタ機能Aの状態をONとする」という意味のルールになる。

【００３９】また、式中の記号「,」は論理「かつ（AND）」の意味で用いられる。例えば、「プリンタ機能Bの状態がON、かつ、プリンタ機能Cの状態がOFFのときに、プリンタ機能Aの状態をONとする」というルールは、

A(ON) ←B(ON), C(OFF).

と記述される。

【００４０】さらに、コンフリクト処理ルール記述ファイル３０１の中には、プリンタドライバＵＩ３０６を更新するための処理を記述することが可能である。推論エンジン３０２がその記述を解釈した時点で、コンフリクトマネージャ３０３の状態変数リスト３０４を介して、プリンタドライバＵＩ３０６の更新処理を直接行うことが可能になっている。

【００４１】例えば、ＵＩ更新の処理として{disable}セットの記述を追加することにより、該当項目のコントロールをdisableする処理（グレイアウト表示するとともに設定不可の状態にする処理）を指示することができる。

【００４２】次に、図８に表示されたプリンタ機能を例に、コンフリクト処理ルールに記述される機能名の具体例を示す。図８に示した[Collate] ８１１に対応する部単位印刷機能、[Group] ８１２に対応するページ単位印刷機能、および[Staple] ８１３に対応するステープル仕上げ機能の機能名はそれぞれ、Collate()、Group()、およびStaple()で示され、引数はONまたはOFFとなる。また、[PrintStyle]欄８０に対応する印刷レイアウト機能は、Layout()で示され、引数は、1-Sided、2-Sided、Bookletのいずれかである。

【００４３】なお、上記した機能名はあくまでプリンタという制御対象に対する一例であって、制御対象に応じてその他の機能名が定義することができることはいうまでもない。

【００４４】図６は、以上の例に従って記述されたコンフリクト処理ルール記述ファイル３０１の一例である。

【００４５】(1)は、[Booklet Printing]８０３がチェックされたことでLayout(BOOKLET)となったときにStaple(OFF)とするルールである。後続行には{disable}が記述されているので、このルールが適用されると、コントロール[Staple]８１３はグレイアウトされることになる。

【００４６】(2)は、[Collate] ８１１がチェックされたことでCollate(ON)となったときにStaple(OFF)とするルール、(3)は、[Group] ８１２がチェックされたことでGroup(ON)となったときにStaple(OFF)とするルールである。

【００４７】(4)は、[Booklet Printing]８０３がチェックされたことでLayout(BOOKLET)となったときにCollate(ON)とするルールであり、このルールが適用されると、次の行の{disable}の記述によってコントロール[collate]８１１がグレイアウトされる。

【００４８】(5)は、[Booklet Printing]８０３がチェックされたことでLayout(BOOKLET)となったときにGroup(OFF)とするルールであり、このルールが適用されると、次の行の{disable}の記述によってコントロール[Group]８１２がグレイアウトされる。

【００４９】このように、各機能の状態によって適用されるルールが記述されている。

【００５０】（プリンタドライバＵＩ制御モジュール２０４２の処理の内容）以下、図５のフローチャートを用いて、コンフリクト処理を含むプリンタドライバＵＩ制御モジュール２０４２の処理について、詳しく説明する。このフローチャートは、（１）ＵＩを介して設定変更要求が発生した場合のコンフリクト処理および、（２）ＵＩを介さずに、他のアプリケーションなどから設定変更要求が発生した場合のコンフリクト処理、を含んでいる。以下、両者の処理を別々に説明する。

【００５１】（１）ＵＩを介して設定変更要求が発生した場合のコンフリクト処理

プリンタドライバＵＩ制御モジュール２０４２の処理は、ユーザがキーボードコントローラＫＢＣ５等を用いて、印刷設定画面を開く指示をすることで始まる。ユーザが印刷設定画面を開くよう指示すると、上記したとおり、ＯＳ２０５の管理の下、ＲＡＭ２に印刷処理関連プログラム２０４がロードされる。

【００５２】印刷処理関連プログラム２０４がＲＡＭ２にロードされると、まず、印刷設定画面を開くための初期化処理として、推論エンジン３０２は、コンフリクト処理ルール記述ファイル３０１をコンフリクトマネージャ３０３を介して、ＲＡＭ２に読み込む（ステップＳ５０１）。

【００５３】続いて、コンフリクトマネージャ３０３が利用する状態変数リスト３０４を作成、初期化する（ステップＳ５０２）。

【００５４】このステップでは、まず、各状態変数の領域を確保しメソッドやフラグを設定する。なお、ＵＩを開かずに初期化する場合は、ＵＩ更新メソッドを無効にしたり、フラグでＵＩ非オープンであることを保持する等の初期化がなされることになる。

【００５５】コンフリクト処理ルール記述ファイル３０１に記述されるすべてのプリンタ機能名は、コンフリクトマネージャ３０３の内部にインクルードされた状態変数リスト３０４にそれぞれ、状態変数を持っている。この状態変数の値は、プリンタドライバＵＩ３０６で使用される内部構造体３０５の対応するメンバの値と連動し

ている。各プリンタ機能名の状態変数の初期値は、その内部構造体３０５のメンバの値となる。このメンバ値をメソッドを通して取得し、状態値にセットする。

【００５６】例えば、状態変数リスト３０４に登録されているプリンタ機能Aのメソッドは、int cAを参照し、そのメンバ値０を取得して対応するプリンタ機能Aの状態値をOFFとする。同様に、プリンタ機能Bの状態値はON、プリンタ機能Cの設定値はOFFとなる。すなわち、

A　OFF
B　ON
C　OFF

となる。

【００５７】その後、設定全体の排他制御をプログラムによって行い、他の項目の設定に依存して利用できなくなっている項目は、対応するプリンタ機能のステータスを＜無効＞とする。ステータスが＜無効＞となっている項目は、ＵＩを表示する場合は該当するコントロールがグレイアウト表示され、ＵＩを表示しない場合はその項目の設定変更要求は受理されなくなる。

【００５８】具体的に、図８に示したＵＩを開く際の初期化処理を例にとって説明する。まず、図７に示す内部構造体中のプリンタ機能Layout、Collate、Group、およびStapleの状態変数を用意して、それぞれに各種メソッドをセットする。次に、各項目の内部構造体取得メソッドを通して内部構造体の設定値からメンバ値を取得し、状態変数値は以下のように初期化される。

Collate　OFF
Group　ON
Staple　OFF
Layout　1-SIDED

【００５９】次に、無効となる項目をチェックし、その項目の状態変数のステータスを＜無効＞にする。この例では、無効となる項目はないものとする。以上でステップＳ５０２の状態変数の初期化処理が完了する。

【００６０】続いて、ＵＩの処理が必要かどうか、すなわちＵＩを表示しているかどうかを判定する（ステップＳ５０３）。判定には呼び出し元を判断したり専用のフラグを用いてもよい。あるいは、状態変数のＵＩ更新メソッドの有無や状態変数内のフラグを利用してもよい。ＵＩをオープンする場合は、状態変数を参照しながらプリンタドライバＵＩのオープンのために必要な初期化処理を行い、ＵＩをオープンする（ステップＳ５０４）。ここでは、設定値に応じた表示を行ったりステータスが＜無効＞の項目はグレイアウト処理を行ったりする。

【００６１】プリンタドライバＵＩがオープンされた後は、OSより送られてくるイベントの取得とその処理を繰り返す（ステップＳ５０５）。ステップＳ５０５で取得したイベントがユーザがプリンタドライバＵＩ上の設定項目を変更したイベントであるかどうかの判定を行う（ステップＳ５０６）。ｙｅｓの場合には、ステップＳ５０７に進み、さらに変更要求が有効な項目かどうかの判定を行う。なお、ＵＩを表示している場合は無効な項目はグレイアウトされているため無効な項目からの入力はないので、この判定は必ず成功する。有効な項目と判定された場合は、ステップＳ５０８に進み、ステップＳ５０１で読み込んだコンフリクト処理ルールを適用する。

【００６２】ここで、取得したイベントがユーザによる設定変更要求であった場合のステップＳ５０８の一例として、図８に示す[Print Style]欄８０における、片面印刷（1-Sided Printing）８０１から製本印刷（Booklet Printing）８０３に変更するものであった場合について説明する。

【００６３】プリンタ機能名Collate、Group、Staple、Layoutの状態変数の設定変更要求前における値は、図７に示した状態、すなわち、以下のようになっていた。

Collate　OFF
Group　ON
Staple　OFF
Layout　1-SIDED

【００６４】ユーザの変更要求がLayoutを１－SidedからBookletに変更するものであるので、Layoutの状態変数の内容が変更されて、各プリンタ機能名の状態変数の値は次のようになる。

Collate　OFF
Group　ON
Staple　OFF
Layout　BOOKLET

【００６５】すると、プリンタドライバＵＩ３０６はコンフリクトマネージャ３０３をコールし、続いて推論エンジン３０２がコールされて、コンフリクト処理ルール記述ファイル３０１を参照してコンフリクトチェックが始まる。すなわち、コンフリクト処理ルール記述ファイル３０１の各ルールの適用／不適用をチェックしていく。

【００６６】まず、ルール(1)が適用されることになるから、StapleはOFFのままコントロールはdisable、すなわちグレイアウト表示によって設定不可の状態にされる。

【００６７】また、ルール(4)、(5)も適用され、Collateの値はOFFからONへ、Groupの値はONからOFFへ変更され、次のように各コントロールはdisableとなる。

Collate　ON (disable)
Group　OFF (disable)
Staple　OFF (disable)
Layout　BOOKLET

【００６８】以上で、ステップＳ５０８の、推論エンジン３０２でのコンフリクト処理ルールの適用が終了し、状態変数リスト３０４の更新が完了する。

【００６９】続いて、ＵＩの処理が必要か否かを判断し

（ステップＳ５０９）、ＵＩの処理が必要な場合には、ステップＳ５１０で、更新が必要なコントロールについて各項目の状態変数のＵＩ更新メソッドを通してＵＩの更新処理を行う。ここでは設定値の更新や、ステータスが＜無効＞となった項目のグレイアウト処理等を行う。

【００７０】以上の処理により、上記の例では、LayoutがI-Sided PrintingからBooklet Printingに設定が変更されたことで、CollateがOFFからONへ、GroupがONからOFFへと変更され、Collate, Group, Staple がそれぞれdisable、すなわちグレイアウト表示となり、図８のＵＩは図１０に示すとおりに更新される。その後、ステップＳ５０５に戻り処理を繰り返す。

【００７１】ステップＳ５０６において、取得したイベントが設定変更要求でないと判断した場合には、ステップＳ５１１に進み、処理の終了要求か、すなわち、ＵＩのクローズ要求かどうかの判別を行う。クローズ要求であった場合には、ステップＳ５１２に進み、状態変数の設定値をメソッドを通して内部構造体に反映させる。続くステップＳ５１３では、ＵＩ処理が必要か否かを判断し、ｙｅｓの場合にはステップＳ５１４で、ＵＩをクローズして、全ての処理を終了する。ステップＳ５１１で、取得したイベントがクローズ要求ではなかった場合には、ステップ５０５に戻って処理を繰り返すことになる。

【００７２】以上の処理は、ＵＩがクローズされるまで、繰り返し実行される。ＵＩがクローズされると処理は全て終了し、本実施形態における印刷処理関連プログラムの処理も終了し、RAM２からはＯＳ２０５の機能により消去される。

【００７３】（２）ＵＩを介さずに、他のアプリケーションなどから設定変更要求が発生した場合のコンフリクト処理

ＵＩを開く操作のかわりに、他のアプリケーションからの設定変更要求の受付により処理がスタートする。プリンタドライバＵＩ３０６が他アプリケーションからの要求を受け付けるインタフェースとしての働きを持つことになる他は、上述した（１）の例によりＵＩを開く場合と構造は同じである。まず、ＵＩ表示時と同様にコンフリクトマネージャ３０３が起動され、コンフリクト処理ルール記述ファイル３０１が読み込まれる（ステップＳ５０１）。

【００７４】次に、ステップＳ５０２で状態変数リスト３０４を作成、初期化するが、ＵＩ表示が不要なのでＵＩ更新メソッドは不要となる。また、必要があればフラグにＵＩ非表示であることをセットしておいてもよい。設定値の初期化と設定全体の排他制御はＵＩ表示時と同じである。なお、ステータスに＜無効＞をセットされた項目は変更要求を受け付けないので、ＵＩ上でグレイアウトされている項目が変更操作を受け付けないのと同等の効果を持つことになる。ここまででＵＩ表示時と同一の状態変数リストが完成する。

【００７５】続いて、ステップＳ５０３にてＵＩ処理が必要であるか否かを判定するが、本例ではＵＩを表示していないので、ステップＳ５０４のＵＩオープン処理はスキップされて、ステップＳ５０５に進む。

【００７６】ステップＳ５０５で、イベントの取得とその処理を繰り返す。ＵＩ表示時はユーザによってなされたＵＩ操作がOSを通してイベントとして渡されるが、ＵＩ非表示時は他のアプリケーションからの要求がイベントとして渡されることになる。ステップＳ５０５で取得したイベントが設定変更要求であるか否か、すなわち項目と設定値の組合せであるか否かを判定し（ステップＳ５０６）、ｙｅｓの場合にはステップＳ５０７に進み、対応する状態変数のステータスを参照して、その項目が有効であるかどうかを判定する。

【００７７】ここで、無効と判断された場合は、ＵＩ表示時であればグレイアウトされるコントロールと同等であることを意味するので、その変更要求は破棄されてステップＳ５０５に戻る。他方、有効と判断されれば、その変更要求項目を起点としてコンフリクト処理を起動し、ＵＩ表示時と同様に処理を進め、状態変数を更新する（ステップＳ５０８）。次のステップＳ５０９（ＵＩ処理の要否判断）ではｎｏと判断されるので、ステップＳ５１０のＵＩ更新処理を行うことなくステップＳ５０５に戻る。

【００７８】このようにプリンタドライバ２０４１およびコンフリクトマネージャ３０３を起動している間に、ステップＳ５０５からＳ５１０の処理を他のアプリケーションから設定変更要求があるかぎり繰り返す。そして変更要求項目がなくなり、終了要求イベントが送られてきたところで（ステップＳ５１１）、ＵＩ表示時と同様に内部構造体３０５の更新を行う（ステップＳ５１２）。なお、本例では、ステップＳ５１３のＵＩ表示の判定ではｎｏと判断されて、ステップＳ５１４のＵＩクローズ処理はスキップされることになる。

【００７９】以上説明した処理によれば、ＵＩを開いてＵＩ上で設定変更した場合とまったく同一の内部構造体を得ることができる。

【００８０】ところで、図５のステップＳ５０２は、メソッドと設定値をセットした後、プログラムによってステータスの判断を行うものであったが、このステップでもコンフリクト処理ルールを適用する例を図９に示すフローチャートを用いて説明する。

【００８１】ここでは、内部構造体３０５の初期状態が図１１に示すとおりであり、この他に、図１２に示す起点リスト、および図６のコンフリクト処理ルールを用いることとする。

【００８２】まず、状態変数の初期設定として、状態変数領域をＲＡＭ２の空き領域２０２（図２を参照）を確保するとともに、メソッドを設定し、メソッドを通して

内部構造体３０５より初期値を取得する一連の処理を行う（ステップＳ９０１）。その結果、状態変数は以下のように初期化される。

Collate ON
Group OFF
Staple OFF
Layout BOOKLET

【００８３】次に、図１２の起点リストを読み込み（ステップＳ９０２）、最初の項目であるLayoutを取得し、その設定値であるBOOKLETと組み合わせた[Layout, BOOKLET]を最初の起点とする（ステップＳ９０３）。まず、その起点となる項目のステータスを評価して（ステップＳ９０４）、有効であればコンフリクト処理を開始し、前記設定変更要求を起点としてコンフリクト処理ルールを適用する。起点を右辺にもつルールとして、まず図６のルール(1)が適用され、StapleはOFF、ステータスはdisableとなる。同様に、ルール(4)および(5)が適用され、状態変数は以下のように更新される。

Collate ON (disable)
Group OFF (disable)
Staple OFF (disable)
Layout BOOKLET

【００８４】これで起点リスト中のLayoutに関するコンフリクト処理が終了する。なお、ステップＳ９０４にて無効な項目と判定された場合は、ステップＳ９０５のコンフリクト処理をスキップする。

【００８５】続いて、起点リストの次の項目であるCollateを取得し、その設定値と合わせた[Collate, ON]を次の起点とする（ステップＳ９０６）。起点の有無のチェック（ステップＳ９０７）を経て、再びステップＳ９０４からの処理に戻る。ステップＳ９０４では今度の起点Collateのステータスチェックを行うが、ルール(4)が適用されたことによってCollateのステータスはdisableであるため、その変更要求は破棄される。

【００８６】同様に、起点リスト中のGroup、Stapleについても順次処理を行い、ステップＳ９０７の判定において起点リスト中のすべての項目が終了したと判断したところで処理を終了となる。ここまでの処理で状態変数は以下のように初期化され、不整合の排除とステータスの初期化が完了する。

Collate ON (disable)
Group OFF (disable)
Staple OFF (disable)
Layout BOOKLET

【００８７】このように、ＵＩ上に表示する項目のうち、他の項目と排他関係が存在する項目をすべて起点リストに列挙しておくことで状態変数全体のステータス判定を行う。また、優先順位の高い項目から起点リストに記載することで従属関係があるものの処理も可能であり、途中で無効となった項目はコンフリクトチェックをスキップすることで処理の効率化を図ることも可能である。

【００８８】上述した実施形態では、ステップＳ５１２で処理終了時に内部構造体の更新を行っているが、その更新のタイミングはこの限りではなく、例えば、ステップＳ５０８の後で状態変数を変更する度に内部構造体を更新するようにしてもよい。

【００８９】また、上述した実施形態では、ステップＳ９０３で、項目と設定値のセットを起点としているが、かわりに項目のみを起点とし、コンフリクトマネージャ側で該当する設定値を補完した後に処理を遂行するようにしてもよい。

【００９０】また、図９の処理を図５のステップＳ５０２部分の詳細として説明したが、かわりに処理全体を、状態変数の初期化までのステップと、その後の内部構造体に書き戻すステップとの組合せにすることで、設定変更は行わずに単に内部構造体のデータ全体の整合処理を行うことも可能である。

【００９１】以上説明した実施形態によれば、ＵＩを介さずに入力されるデータを更新する処理、およびデータ全体の整合を取る処理において、ＵＩのコンフリクト処理をそのまま利用することができる。したがって、両者の処理のための専用プログラムを作成する煩わしさや条件の不一致や漏れによって生じる不具合から解放され、品質の高い整合処理を低コストで実現できるという効果がある。

【００９２】

【他の実施形態】なお、上述した実施形態では、プリンタ装置に対して、コンフリクト処理を含むＵＩ制御を行う例について説明したが、本発明は、プリンタ装置に限らず、ディジタルカメラ、ディジタルレコーダ、イメージスキャナ等の周辺装置、制御機器の他、モデムやルータといったネットワーク関連機器にも適用可能であることはいうまでもない。これらの複数の機器から構成されるシステムに適用することも可能である。

【００９３】上述したように、本発明の目的は、前述した実施形態の機能を実現するソフトウェアのプログラムコードを記録した記憶媒体（または記録媒体）を、システムあるいは装置に供給し、そのシステムあるいは装置のコンピュータ（またはＣＰＵやＭＰＵ）が記憶媒体に格納されたプログラムコードを読み出し実行することによっても、達成される。この場合、記憶媒体から読み出されたプログラムコード自体が前述した実施形態の機能を実現することになり、そのプログラムコードを記憶した記憶媒体は本発明を構成することになる。また、コンピュータが読み出したプログラムコードを実行することにより、前述した実施形態の機能が実現されるだけでなく、そのプログラムコードの指示に基づき、コンピュータ上で稼働しているオペレーティングシステム（ＯＳ）などが実際の処理の一部または全部を行い、その処理に

よって前述した実施形態の機能が実現される場合も含まれることは言うまでもない。

【００９４】さらに、記憶媒体から読み出されたプログラムコードが、コンピュータに挿入された機能拡張カードやコンピュータに接続された機能拡張ユニットに備わるメモリに書込まれた後、そのプログラムコードの指示に基づき、その機能拡張カードや機能拡張ユニットに備わるＣＰＵなどが実際の処理の一部または全部を行い、その処理によって前述した実施形態の機能が実現される場合も含まれることは言うまでもない。

【００９５】本発明を上記記憶媒体に適用する場合、その記憶媒体には、先に説明した図５または、図５および図９のフローチャートに対応するプログラムコードが格納されることになる。

【００９６】

【発明の効果】以上説明したように、本発明によれば、ユーザインタフェースを介して入力されるデータに対するコンフリクト処理を行うためのモジュールを、ユーザインタフェースを介さずに入力されるデータに対するコンフリクト処理とデータ全体の整合処理とで共用可能にして、各処理ごとの専用のコンフリクト処理モジュールを作成する必要をなくし、もって低コストで高品質なユーザインタフェース制御装置および方法を提供することができる。

【図面の簡単な説明】

【図１】実施形態に係る印刷処理システムのブロック構成図である。

【図２】実施形態におけるＲＡＭ２のメモリマップを示す図である。

【図３】実施形態におけるプリンタドライバＵＩ制御モジュールの概略構成図である。

【図４】実施形態におけるプリンタドライバＵＩ制御モジュールで扱われるデータの関連を説明する模式図である。

【図５】実施形態におけるプリンタドライバＵＩ制御モジュールの処理を示すフローチャートである。

【図６】実施形態におけるコンフリクト処理ルール記述ファイルの一例を示す図である。

【図７】実施形態における内部構造体の一例を示す図である。

【図８】実施形態におけるプリンタドライバＵＩによる印刷設定画面の一例を示す図である。

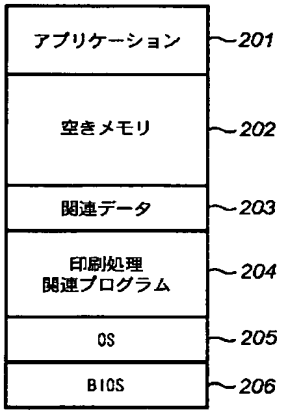
【図９】実施形態における状態変数リストの作成後のコンフリクト処理ルールの適用処理の例を示すフローチャートである。

【図１０】実施形態におけるプリンタドライバＵＩによる印刷設定画面の一例を示す図である。

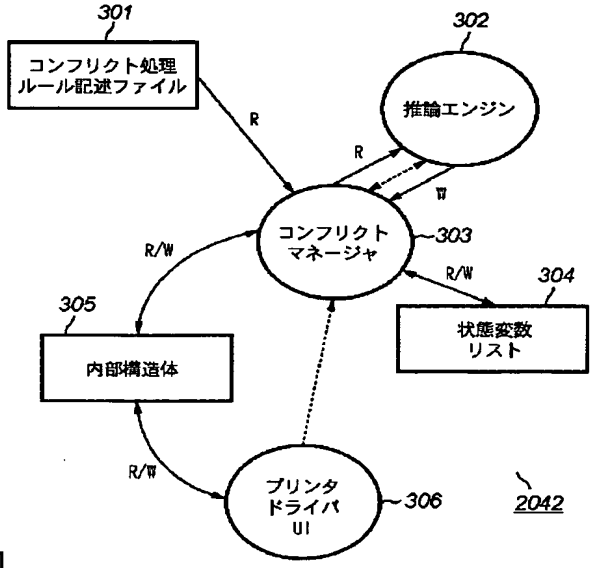
【図１１】実施形態における内部構造体の一例を示す図である。

【図１２】実施形態における起点リストの一例を示す図である。

【図２】




【図３】




【図６】

```
Staple(OFF)←Layout(BOOKLET).          (1)
    {disable}
Staple(OFF)←Collate(ON).               (2)
Staple(OFF)←Group(ON).                 (3)

Collate(ON)←Layout(BOOKLET).           (4)
    {disable}
Group(OFF)←Layout(BOOKLET).            (5)
    {disable}
      :
```

【図１１】

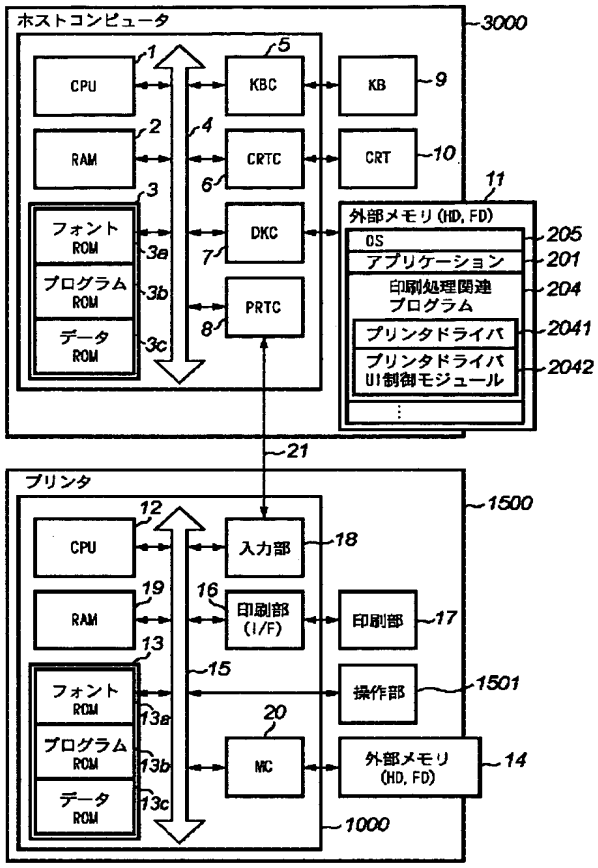
2042

| | | |
|---|---|---|
| int | cLayout | Booklet |
| int | cCollate | 1 |
| int | cGroup | 0 |
| int | cStaple | 0 |
| | : | |

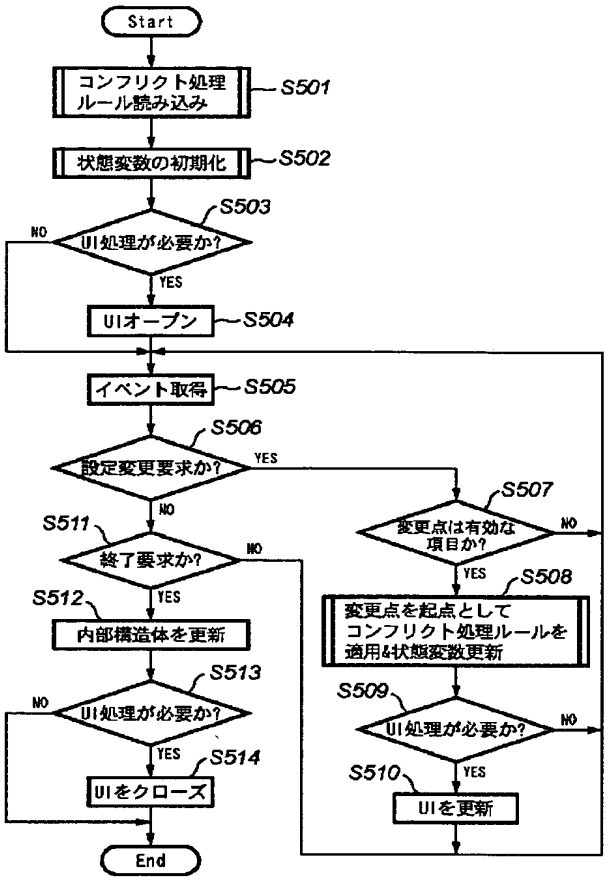
【図７】

| | | |
|---|---|---|
| int | cLayout | 1Sided |
| int | cCollate | 0 |
| int | cGroup | 1 |
| int | cStaple | 0 |
| | : | |

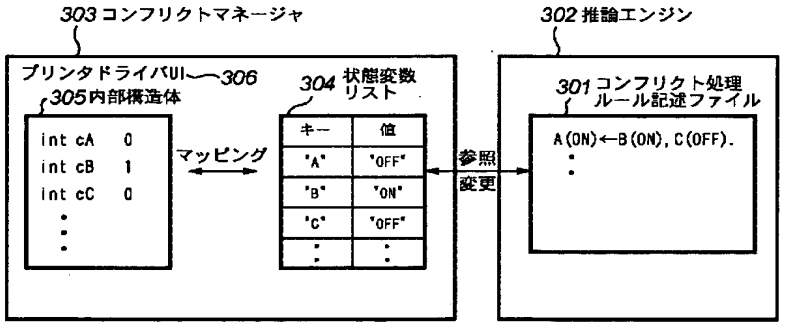
【図１】


ホストコンピュータ
3000
CPU
1
RAM
2
フォント ROM
3
3a
プログラム ROM
3b
データ ROM
3c
KBC
5
CRTC
4
6
DKC
7
PRTC
8
KB
9
CRT
10
外部メモリ(HD, FD)
11
OS
205
アプリケーション
201
印刷処理関連プログラム
204
プリンタドライバ
2041
プリンタドライバUI制御モジュール
2042
21
プリンタ
1500
CPU
12
RAM
19
フォント ROM
13
13a
プログラム ROM
13b
データ ROM
13c
入力部
18
印刷部(I/F)
16
印刷部
17
15
操作部
1501
NC
20
外部メモリ(HD, FD)
14
1000

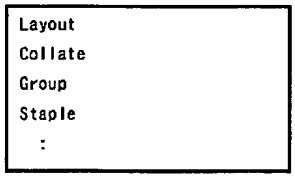
【図５】


Start
コンフリクト処理ルール読み込み
S501
状態変数の初期化
S502
S503
UI処理が必要か?
NO
YES
UIオープン
S504
イベント取得
S505
S506
設定変更要求か?
YES
NO
S507
変更点は有効な項目か?
NO
YES
S508
変更点を起点としてコンフリクト処理ルールを適用&状態変数更新
S509
UI処理が必要か?
NO
YES
S510
UIを更新
S511
終了要求か?
NO
YES
S512
内部構造体を更新
S513
UI処理が必要か?
NO
YES
S514
UIをクローズ
End

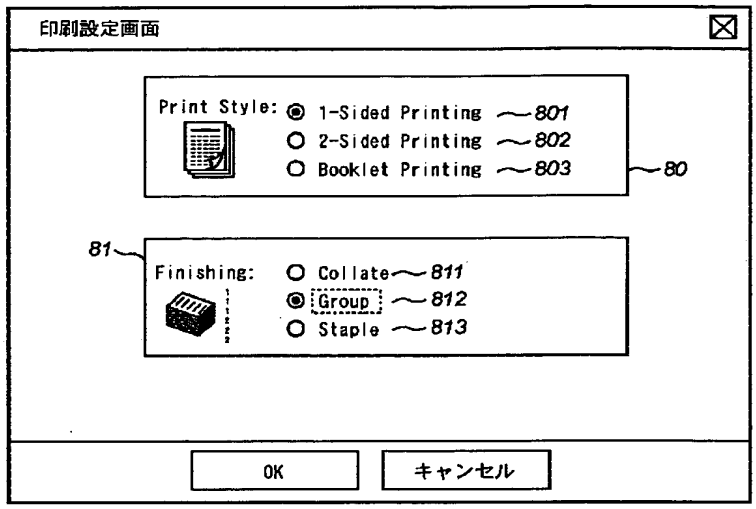
【図４】


303 コンフリクトマネージャ
302 推論エンジン
プリンタドライバUI
306
305 内部構造体
int cA 0
int cB 1
int cC 0
マッピング
304 状態変数リスト
キー
値
"A"
"OFF"
"B"
"ON"
"C"
"OFF"
参照
変更
301 コンフリクト処理ルール記述ファイル
A(ON)←B(ON), C(OFF).

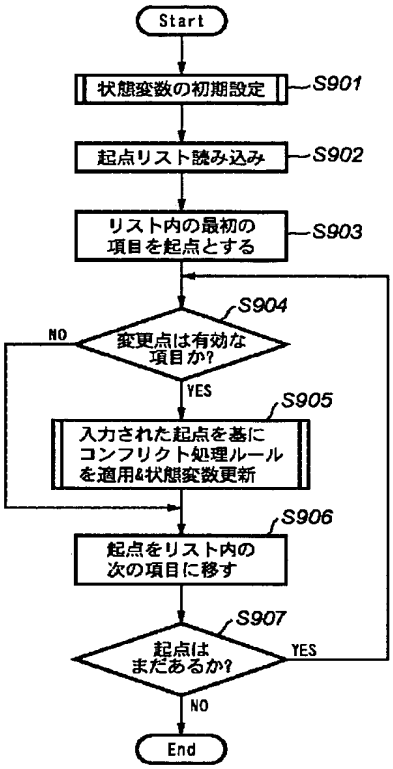
【図１２】


Layout
Collate
Group
Staple

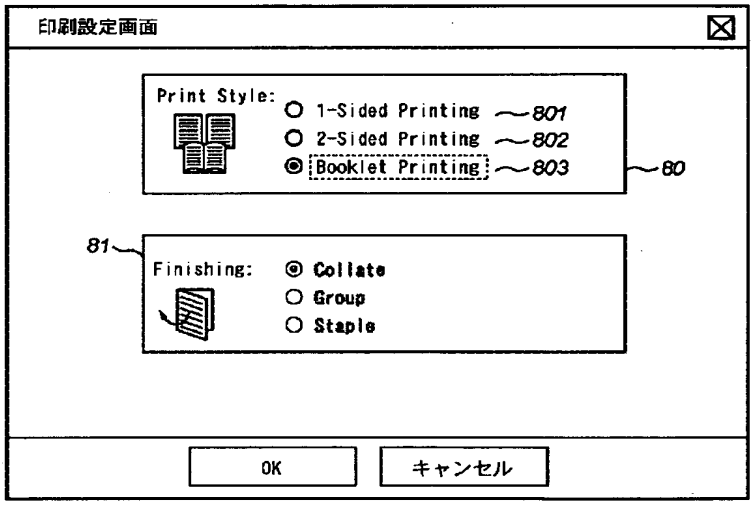
【図８】


印刷設定画面
Print Style:
1-Sided Printing
2-Sided Printing
Booklet Printing
801
802
803
80
81
Finishing:
Collate
Group
Staple
811
812
813
OK
キャンセル

【図９】


Start
状態変数の初期設定
S901
起点リスト読み込み
S902
リスト内の最初の
項目を起点とする
S903
S904
変更点は有効な
項目か?
NO
YES
S905
入力された起点を基に
コンフリクト処理ルール
を適用&状態変数更新
S906
起点をリスト内の
次の項目に移す
S907
起点は
まだあるか?
YES
NO
End

【図１０】


印刷設定画面
Print Style:
1-Sided Printing
2-Sided Printing
Booklet Printing
801
802
803
80
81
Finishing:
Collate
Group
Staple
OK
キャンセル